Capstone Final Report: Chest X-Ray Covid Classification

Problem Statement

It is estimated that 3.6 billion diagnostic x-ray examinations are performed every year in the world[(1)]. The chest x-ray is the most performed diagnostic x-ray examination. A chest x-ray produces images of the heart, lungs, airways, blood vessels and the bones of the spine and chest. A chest x-ray is typically the first imaging test used to help diagnose symptoms such as:

- breathing difficulties
- a bad or persistent cough
- chest pain or injury
- fever[(2)]

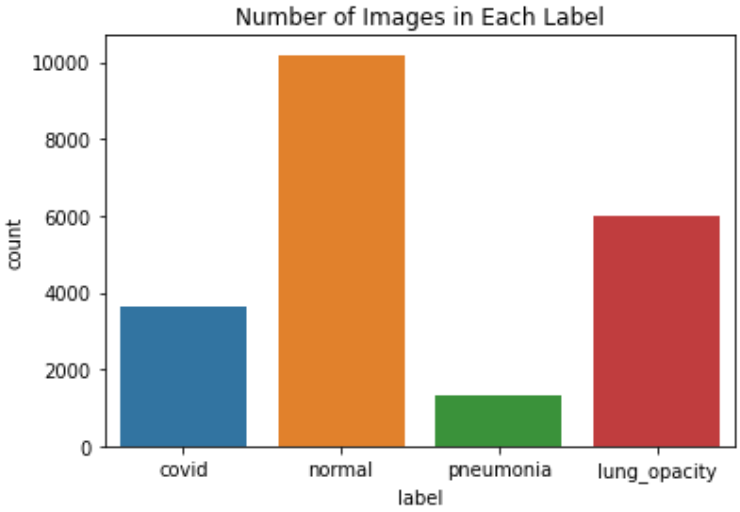
Radiologists' accuracy varies depending on what condition is being diagnosed. For example, pneumonia is accurately diagnosed from a chest x-ray around 85% of the time. Studies have also shown that there is disagreement in the interpretation of x-rays in up to 56% of cases[(3)].

In the United States, national average cost of chest x-ray was $370 in 2010[(4)] with results taking minutes, in emergent cases, to multiple weeks. With so much manpower being used to assess x-rays, around the world, each year, it would be beneficial to use a machine learning model to do the initial assessment and flag which x-rays have abnormalities. Using machine learning would save both time and money for the patients.

Data Wrangling

The images for this project were taken from the COVID-19 Radiography database on Kaggle which can be found here. The data set included metadata files for each of the four image classes: Normal, Viral Pneumonia, Lung Opacity, and COVID. In the metadata files contained four columns: FILE NAME, FORMAT, SIZE, and URL. Since the full file name is a combination of FILE NAME and FORMAT, I created a new DataFrame that took the file names and labels directly from the folders containing the images.

From my new DataFrame, I looked at the distribution of the image labels.



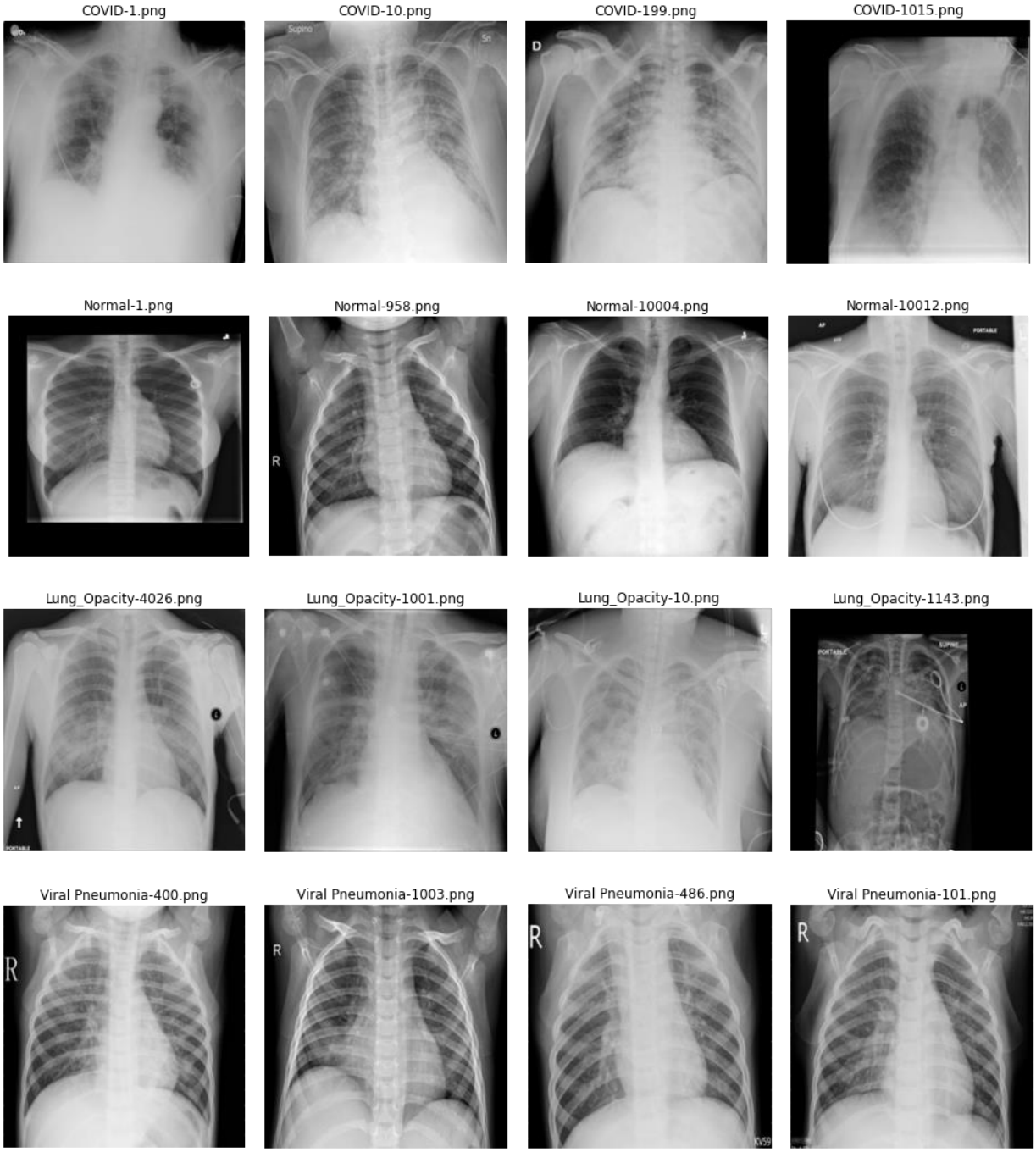
This dataset is unbalanced with only 6% of the images being viral pneumonia cases and normal cases making up 48% of the dataset.

As the final step of data wrangling, I added the image size and image paths to the DataFrame and resized each image to be half of the original size.

The full Jupyter notebook – Data_Wrangling_and_EDA – can be found here.

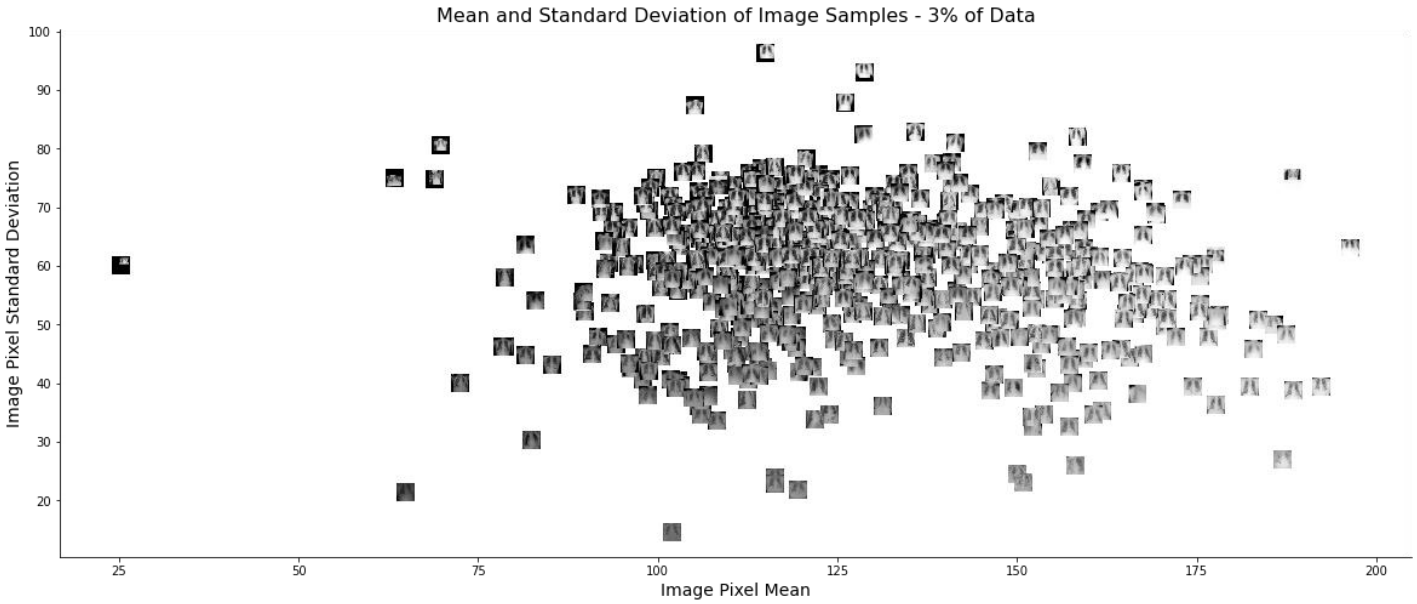
Exploratory Data Analysis (EDA)

During EDA, I first looked at a sample of images from each of the four classes.



As you can see in the sample of images, the range of pixel values can be quite different image-to-image. To get an idea of the range of pixel values, I looked at a sample of histograms of pixel values. I then calculated some descriptive statistics for the pixel values for each image and added them to the

DataFrame. For each image, I calculated minimum, maximum, mean, and standard deviation of the pixel values. I then plotted 3% of the data's mean vs standard deviation using the images as points.



This plot is great for visualizing what these descriptive statistics mean. Images with mean values closer to 0 are darker. While images with a larger mean value, have more white. The images with higher standard deviation generally have darker background with the bones and organs appearing brighter white. The images with lower standard deviation appear to be more uniformly grey.

Next, I looked at applying histogram equalization versus adaptive equalization on the images. Upon closer look, it appeared that adaptive histogram equalization had already been applied. Since adaptive equalization is good for improving the local contrast and enhancing the definitions of edges in each region of an image, I chose not to do any further equalization.

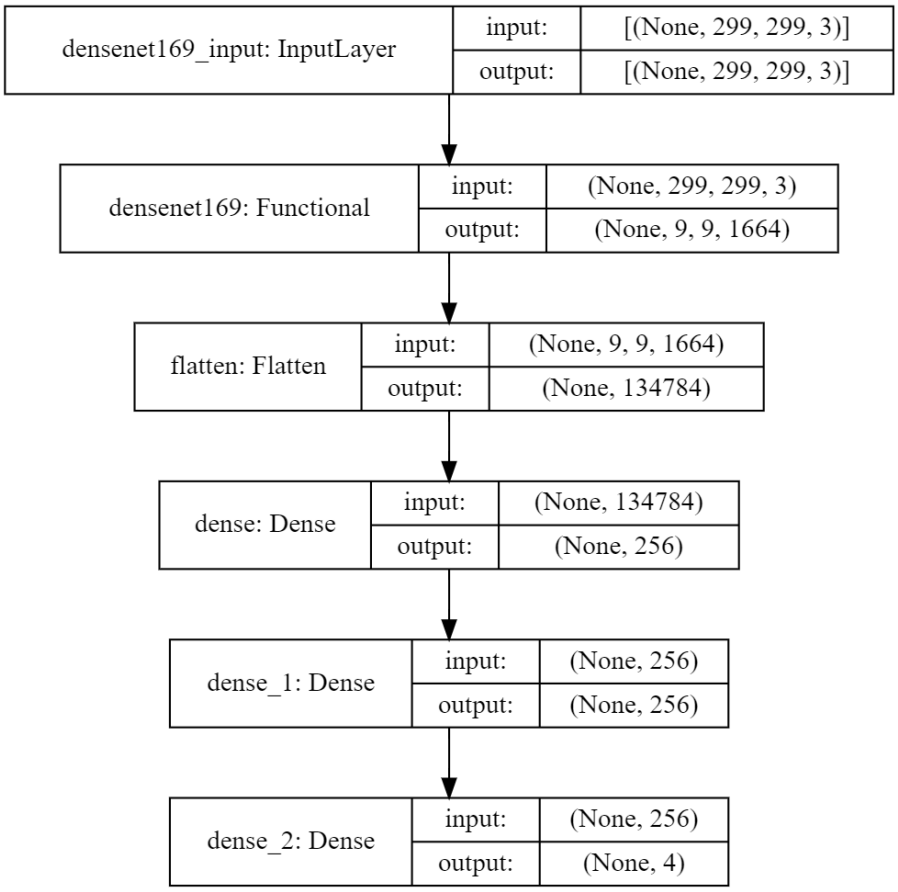
Finally, I looked at the center of each image. I found that the lung opacity images had the greatest spread of center points. In contrast, the pneumonia images were the most centered of the image set. I then re-centered each image by calculating how far, in each direction, the image was from the center and then shifting the image by that number of pixels.

The full Jupyter notebook – Data_Wrangling_and_EDA – can be found here.

Data Preprocessing and Training

Before modeling the data, I first separated the images into 70% training and 30% validation sets using the images' file names. I then set up some functions to speed up the training process.

I used transfer learning to train two models: DenseNet-169 and DenseNet-121. The purpose of DenseNet is to train a convolutional neural network that classifies images from the CIFAR-10 database in Keras. To avoid the computational cost of training a CNN from scratch, DenseNet-169 and DenseNet-121 were pre-trained from ImageNet. The structure of the DenseNet-169 CNN is below:



The Jupyter notebook – Preprocessing_and_Modeling – can be found here.

Model Selection and Prediction on Validation Data

The DenseNet-169 model performed slightly better than the DenseNet-121 model in both accuracy and loss. When the model was then used on the validation set, it performed with just over 92% categorical accuracy.

The Jupyter notebook – Preprocessing_and_Modeling – can be found here.

Future Work

For future work, I would like to work on a model for classifying all medical imaging, such as:

- X-rays
- CT (computed tomography) scans
- MRI (magnetic resonance imaging)
- Ultrasounds
- Positron-emission tomography (PET) scans